DOCUMENT RESUME

ED 079 399 TM 003 002

AUTHOR van Thillo, Marielle
TITLE Computer Subroutines for Analytic Rotation by Two Gradient Methods.
INSTITUTION Educational Testing Service, Princeton, N.J.
REPORT NO ETS-RM-73-11
PUB DATE Jun 73
NOTE 44p.

EDRS PRICE MF-$0.65 HC-$3.29
DESCRIPTORS Comparative Analysis; *Computer Programs; *Factor Analysis; *Matrices; *Oblique Rotation; *Orthogonal Rotation; Statistical Analysis

ABSTRACT

Two computer subroutine packages for the analytic rotation of a factor matrix, A(p x m), are described. The first program uses the Fletcher (1970) gradient method, and the second uses the Polak-Ribiere (Polak, 1971) gradient method. The calculations in both programs involve the optimization of a function of free parameters. The result is a transformation matrix, T(m x m), giving us the rotated factor matrix, AT(p x m). The advantage of the Fletcher method, a modification of the Fletcher Powell (1963) method, is that convergence is more rapid than with the Polak-Ribiere method. However, since Fletcher's method builds up an inverse Hessian matrix and the Polak-Ribiere method does not, more computer storage has to be used in the first program than in the second. Therefore, when m is large, the Polak-Ribiere method will have to be used because of storage considerations. Listings of subroutine ROTATE with subprograms using both methods are provided in an appendix. Details of the author's experience with both gradient methods are also given. (Author/KM)

FILMED FROM BEST AVAILABLE COPY

ED 079399

TM 003 002

RM-73-11

RESEARCH MEMORANDUM

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

COMPUTER SUBROUTINES FOR ANALYTIC ROTATION BY TWO GRADIENT METHODS

Marielle van Thillo

This Memorandum is for interoffice use. It is not to be cited as a published report without the specific permission of the author.

Educational Testing Service
Princeton, New Jersey
June 1973

Computer Subroutines for Analytic Rotation by Two Gradient Methods

1. INTRODUCTION

We shall describe two computer subroutine packages for the analytic rotation of a factor matrix, $A(p \times m)$. The first program uses the Fletcher (1970) gradient method, and the second program uses the Polak-Ribière (Polak, 1971) gradient method. The calculations in both programs involve the optimization of a function of free parameters, x_i; $m(m-1)/2$ in the case of orthogonal rotation and $m(m-1)$ in the case of oblique rotation. The result is a transformation matrix, $T(m \times m)$, giving us the rotated factor matrix, $AT(p \times m)$.

The advantage of the Fletcher method, a modification of the Fletcher-Powell (1963) method, is that convergence is more rapid than with the Polak-Ribière method. However, since Fletcher's method builds up an inverse Hessian matrix and the Polak-Ribière method does not, more computer storage has to be used in the first program than in the second. Therefore, when m is large, the Polak-Ribière method will have to be used because of storage considerations. For listings of the program packages see Appendix A.

The theory, formulae and derivations are described in a paper by M. W. Browne (1973). This paper also gives details concerning our experience with both gradient methods.

2. PROGRAM OPTIONS

Various options are available to the user, some of which will be described in some detail in this section.

The user can request an orthogonal or an oblique rotation; he can choose his criterion for solution from the Crawford-Ferguson (1970) family; he can Kaiser normalize the input matrix, A , before rotation or not; he can print technical output from the minimization or not. Furthermore, the results from a previous call to the subroutine package for orthogonal rotation can be used as a starting point for an oblique rotation. A starting vector of free parameters with respect to which the function is to be minimized can either be defined within the subroutine or be read in as data by the user. The latter is especially useful if the time limit has been exceeded during execution before convergence has been reached. With little loss of information the run can be continued reading in as a starting point the vector of free parameters, returned in the calling sequence, when execution was terminated.

Some of these options need further explanation such as the user's choice of criterion for solution. The family of criteria for simple structure, proposed by Crawford and Ferguson (1970), are dependent on parameters κ_1 and κ_2 :

$$f = \kappa_1 \, tr[D_\eta^2] + \kappa_2 \, tr[D_\gamma^2] - tr[\Lambda' \Lambda^3]$$

where

$$\Lambda = AT$$

$$D_\eta = \text{Diag}(\Lambda\Lambda')$$

$$D_\gamma = \text{Diag}(\Lambda'\Lambda)$$

$$[\Lambda^3]_{ij} = \lambda_{ij}^3 \quad .$$

By specifying different values for κ_1 and κ_2 the subroutine determines the criterion for solution, e.g., if $\kappa_1 = p$ and $\kappa_2 = 0$ the quartimax solution is computed (see Browne, 1973). The program ensures that $\kappa_1 + \kappa_2 = 1$ by scaling κ_1 and κ_2 internally, i.e., $\kappa_1/(\kappa_1 + \kappa_2) \rightarrow \kappa_1$ and $\kappa_2/(\kappa_1 + \kappa_2) \rightarrow \kappa_2$.

After obtaining an orthogonally or obliquely rotated factor matrix, AT , the user can call the subroutine package again and request a 'clean-up' rotation of AT . This involves an orthogonal or oblique rotation to a least squares fit to a partially specified target. The target is constructed implicitly by the program and has specified zero elements corresponding to 'small' values of AT . The function, f , minimized is the sum of squares of all elements less in absolute value than η times the largest magnitude in each row, i.e.,

$$f = \sum_{[AT]_{ij}^2 < \eta^2 q_i} \sum [AT]_{ij}^2 \quad ; \qquad i = 1,p \text{ and } j = 1,m$$

where

$$q_i = \max_k [AT]_{ik}^2 \quad ; \qquad k = 1,m$$

$$0 < \eta < 1 \quad .$$

In the program $\eta = .5$.

The technical output consists of a series of tables that describe the behavior of the iterative procedure and give various measures of the accuracy of the final solution. Ordinarily users will have little interest in these tables. The tables show the behavior of the iterative procedure under the steepest descent iterations and under the subsequent iterations by the Fletcher method. For interpretation of these tables the reader is referred to Gruvaeus and Jöreskog (1970) and Fletcher (1970). If something goes wrong with the minimization, so that IND = 2 or 3 (see section 3), these tables may contain valuable information. The inverse of the built-up Hessian matrix is also printed.

The time it took in seconds to complete the minimization as well as the value of the indicator IND (see section 3) is always printed.

5. SUBROUTINE ROTATE USING THE FLETCHER GRADIENT METHOD

Calling Sequence

CALL ROTATE(ITP,MAXITE.ORD,OPR,ORT,OSC,ORS,F,X,D,Y,E.IND.G,H,SC,FMT)

Input Parameters

ITP — determines criterion for solution (see sec. 2).
= 0 for a 'clean-up' rotation.
= 1 for the quartimax solution.
= 2 for the varimax solution.
= 3 for the equamax solution.
= 4 for the parsimax solution.
= 5 for the factor parsimony solution.
= 6 other, i.e., κ_1 and κ_2 are defined by the user.

MAXITE — maximum number of iterations allowed to reach convergence.
If set equal to zero, MAXITE is defined in the program to be 5N (N = number of free parameters).

ORD — logical variable (1 byte)
= F if the starting vector, $\underline{x}$, of free parameters is defined in the program.
= T subroutine ROTATE expects the starting vector, $\underline{x}$, as data preceded by a variable format card.

OPR logical variable (1 byte)

= T if technical output is requested (see sec. 2).

= F otherwise.

ORT logical variable (1 byte)

= T for an orthogonal rotation.

= F for an oblique rotation.

OSC logical variable (1 byte)

= T if the input matrix, A , is row-scaled before rotation (Kaiser normalization).

= F otherwise.

ORS logical variable (1 byte)

= T if the transformation matrix, T , computed in a previous call to ROTATE where the input matrix A was orthogonally rotated is to be used as a starting point in the present call to ROTATE for an oblique or 'clean-up' rotation.

Note: 1) OSC should be the same in both calls.

2) ORS is a dummy parameter if ORT = T and ITP $\neq$ 0 .

3) ORD is a dummy parameter if ORS = T .

4) if ITP = 0 , ORS must be = T .

= F otherwise.

Output Parameters

F — simplicity criterion

X — vector of free parameters; must be dimensioned in the calling program by at least N where $N = m(m - 1)/2$ in the orthogonal case and $N = m(m - 1)$ in the oblique case.

D — column sums of squares of the rotated factor matrix, AT ; must be dimensioned in the calling program by at least N .

Y — vector of communalities; must be dimensioned in the calling program by at least N .

E — rotated factor matrix, AT ; must be dimensioned in the calling program by at least $\max(N(N + 1)/2, pm)$

IND
- = 0 if convergence was reached.
- = 2 if the minimum along a line was not found in 10 trials. The steepest descent direction was then tried but the same thing happened--check your input data.
- = 3 if a positive initial slope was found along a direction--check your input data.
- = 4 if the number of iterations has exceeded MAXITE.
- = 5 if the time limit was exceeded.

Internal Parameters

G,H,SC — must be dimensioned in the calling program by at least N .

FMT — must be dimensioned in the calling program by at least 10.

Parameters Transmitted through COMMON

The following two COMMON blocks have to appear in the calling program:

COMMON/TID/SEC

SEC (input) — is to be defined by the calling program. It is the estimated time in seconds allowed to reach convergence. SEC should be slightly less than the time specified on control cards so that the program will have time to print results up to the point where it ran out of time. (SEC is REAL*4, i.e., 4 bytes.)

COMMON/MN/Q1,Q2,A(400),T(100),AT(400),PP(100),Z(100),P,M,N

Q1,Q2 (input) — κ_1, κ_2 (see sec. 2) have to be defined by the user only if ITP = 6 .

A (input) — input factor matrix to be rotated; should be defined as a vector, reading row-wise.

T,AT (internal) — used for internal calculations.

PP (output) contains the transformation matrix T(m x m) upon return to the program. This matrix is stored as a vector, reading row-wise.

Z (output) will contain the factor correlation matrix in the oblique case. Only the lower triangular part is stored as a vector, reading row-wise.

P (input) integer: number of variables, p ; must be defined in the calling program.

M (input) integer; number of factors, m ; must be defined in the calling program.

N (internal) = m(m - 1)/2 for the orthogonal rotation.
= m(m - 1) for the oblique rotation. N is defined in subroutine ROTATE.

Program Limitations

max. no. of variables p = 40

max. no. of factors m = 10

storage requirements = 78 K (K = 1024 bytes)

machine: IBM 360/65

language: FORTRAN IV-G

precision: double precision is used in floating point arithmetic throughout the program package.

Subprograms Called by ROTATE

FLEPWR, FCTGR, SEARCH, REORDX, STEDE, POINT, CLEAN, PMSL, LAV, ROWSCL, SYMAMU, TIRP, INTPOL, IFMSL, MMSL

Increasing Dimensions

If the number of variables, p , exceeds 40 and/or the number of factors, m , exceeds 10 the dimensions can be increased by making minor changes in subroutines ROTATE and FCTGR:

1) In subroutines ROTATE and FCTGR, COMMON blocks /MN/ and /FUNC/ are dimensioned as follows:

COMMON/MN/Q1,Q2,A(pm),T(m^2),AT(pm),PP(m^2),Z(m^2),P,M,N

COMMON/FUNC/ ... OCNT(pm) where OCNT is type logical (1 byte).

2) In subroutine FCTGR:

DIMENSION X(1),G(1),DH($m^2/2$),DL($m^2/2$)

EQUIVALENCE (DH,Z),(DL,Z($m^2/2+1$))

3) Caution: in 1) and 2) above the following must hold:

$p \leq N$

$p + m \leq m^2$

Disclaimer

Although the program package has been working satisfactorily for all data analyzed so far, no claim is made that it is free of error and no warranty is given as to the accuracy and functioning of the program.

4. SUBROUTINE ROTATE USING THE POLAK-RIBIERE GRADIENT METHOD

The description of subroutine ROTATE using the Polak-Ribiere gradient method is almost identical to that of subroutine ROTATE using the Fletcher gradient method. The minor differences are as follows:

1) The matrix E does not appear in the calling sequence of ROTATE.

2) The rotated factor matrix AT will be returned in array AT (see COMMON block MN) not in array E since there is no E .

3) The default value of parameter MAXITE in the calling sequence is 11N.

4) Storage requirements are 70K.

5) Subprograms called: POLRIB, FCTGR, SEARCH, REORDX, STEDE, POINT, CLEAN, PMSL, LAV, ROWSCL, SYMAMU, TIRP, INTPOL, IFMSL, MMSL. Even though FCTGR, SEARCH, REORDX, STEDE, POINT and CLEAN have the same names as subprograms called by ROTATE using the Fletcher gradient method, they are not the same routines--there are differences. All other subprograms are identical in both program packages.

REFERENCES

BROWNE, M. W. (1973). Gradient methods for analytic rotation. Research Bulletin 73-22. Princeton, N.J.: Educational Testing Service.

CRAWFORD, C. B. & FERGUSON, G. A. (1970). A general rotation criterion and its use in orthogonal rotation. Psychometrika 35, 321-332.

FLETCHER, R. (1970). A new approach to variable metric algorithms. Comp. J. 13, 317-322.

FLETCHER, R. & POWELL, M. J. D. (1963). A rapidly convergent descent method for minimization. Comp. J. 6, 163-168.

GRUVAEUS, G. T. & JÖRESKOG, K. G. (1970). A computer program for minimizing a function of several variables. Research Bulletin 70-14. Princeton, N.J.: Educational Testing Service.

POLAK, E. (1971). Computational Methods in Optimization. New York: Academic Press.

Appendix A

I. Listing of Subroutine ROTATE with Subprograms Using the Fletcher Gradient Method

```
      SUBROUTINE ROTATE (ITP,MAXITE,ORD,OPR,ORT,OSC,ORS,E,X,D,Y,E,IND,G,
     1H,SC,FMT)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(C)
      INTEGER P
      REAL CLOCK,T1,T2,VAC
      COMMON/FUNC/CONST,MM,MM1,ITS,OIN,OCNT(400)
      COMMON/MN/Q1,Q2,A(400),T(100),AT(400),PP(100),Z(100),P,M,N
      DIMENSION       X(1),G(1),Y(1),H(1),D(1),E(1),FMT(1),SC(1)
      CALL SETCLK
      EPS=.10-4*P
      OIN=ORT
      ITS=ITP
      MM=M*M
      MM1=MM+1
      IF(ITP.EQ.0)GO TO 56
      GO TO(5,10,15,20,25,1),ITP
    1 TMP=Q1+Q2
      Q1=Q1/TMP
      Q2=Q2/TMP
      GO TO 30
    5 Q1=P
      Q2=0.DO
      GO TO 1
   10 Q1=P-1
      Q2=1.DO
      GO TO 1
   15 Q1=2*P-M
      Q2=M
      GO TO 1
   20 Q1=P-1
      Q2=M-1
      GO TO 1
   25 Q1=0.DO
      Q2=P
      GO TO 1
   30 N=M*(M-1)
      IF(ORT)N=N/2
      IF(ORS)GO TO 41
      IF(OSC)CALL ROWSCL(P,M,A,SC,0)
      IF(ORD)GO TO 40
      DO 35 I=1,N
   35 X(I)=0.DO
      GO TO 45
   40 READ(5,100)(FMT(I),I=1,10)
      READ(5,FMT)(X(I),I=1,N)
      GO TO 45
   41 L=0
      K=0
      DO 43 I=1,M
      DO 43 J=1,M
      K=K+1
      LL=(J-1)*M+J
      IF(I.EQ.J)GO TO 43
      L=L+1
      X(L)=T(K)/T(LL)
   43 CONTINUE
   45 IF(.NOT.ORT) GO TO 60
      L=0
      CONST=0.DO
      DO 55 I=1,P
      SUM=0.DO
      DO 50 J=1,M
      L=L+1
```

```
   50 SUM=SUM+A(L)**2
   55 CONST=CONST+SUM**2
      CONST=CONST*Q1
      GO TO 60
   56 CONST=.25D0
      L=0
      LL=0
      DO 59 J=1,P
      L=L+1
      TMP=AT(L)**2
      DO 57 I=2,M
      L=L+1
      TMP1=AT(L)**2
      IF(TMP1.GT.TMP)TMP=TMP1
   57 CONTINUE
      Q    =TMP*CONST
      DO 58 I=1,M
      LL=LL+1
      OCNT(LL)=.FALSE.
      IF(AT(LL)**2.GT.Q    )GO TO 58
      OCNT(LL)=.TRUE.
   58 CONTINUE
   59 CONTINUE
   60 T1=CLOCK(VAD)
      IF(MAXITE.EQ.0)MAXITE=5*N
      CALL STEDE(N,X,Y,G,H,D,E,F,EPS,OPR,IND)
      IF(IND.NE.4)GO TO 75
      L=0
      DO 70 I=1,N
      DO 65 J=1,I
      L=L+1
   65 E(L)=0.D0
   70 E(L)=1.D0
      CALL RECROX(X,ORT)
      CALL FLEPWR(N,X,Y,G,H,D,E,F,EPS,OPR,IND,MAXITE)
      IF(OPR)CALL PMSE(N,N,E,20H(5X,I5,5X,10D11.3)  ,4HE    ,1,1,1)
   75 T2=CLOCK(VAD)
      T1=(T1-T2)*1.E-6
      WRITE(6,200)T1,IND
      IF(.NOT.OPR)GO TO 77
      IF(ITP.NE.0)GO TO 77
      WRITE(6,400)
      DO 76 I=1,P
      K=(I-1)*M+1
      J=K+M-1
      WRITE(6,500)(OCNT(L),L=K,J)
   76  CONTINUE
   77 IF(IND.GE.4) WRITE(6,300)(X(I),I=1,N)
      IF(ORT)GO TO 82
      L=MM
      K=M*P
      DO 80 I=1,M
      DO 80 J=1,I
      L=L+1
      K=K+1
   80 F(K)=AT(L)
   82 CALL MMSL(P,M,M,A,T,AT,0,0,1)
      IF(OSC)CALL RCWSCL(P,M,AT,SC,1)
      CALL CLEAN(P,M,AT,T,D,PP,E,Y,Z,ORT)
      RETURN
  100 FORMAT(10A8)
  200 FORMAT(1H0,10X,'TIME=',F8.2//   11X,'IND=',I1)
  300 FORMAT(1H1,10X,'X=',7D15.7/13X,(7D15.7))
  400 FORMAT('OLOCATION OF LARGE(F) AND SMALL(T) ELEMENTS IN AT'//)
  500 FORMAT(10X,80L2)
      END
```

```
      SUBROUTINE FLEPWR (N,X,Y,G,H,D,E,F,EP,O,IND,MAXITE)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      LOGICAL LAV
      COMMON /ESP/ TRIAL,FACT,AA,BB,EPS,DIFF,MAXTRY,ITER,IP
      DIMENSION X(1),Y(1),G(1),H(1),D(1),E(1)
      IP=0
      TRIAL=.100
      IND=0
      MAXTRY=10
      FACT=.100
      AA=.2D+1
      BB=.7500
      EPS=EP
      CALL FCTGR  (X,FO,G)
      IF( O ) WRITE (6,11)
      DO 300 ITER=1,MAXITE
      IF(LAV(N,G,EPS)) GO TO 320
      CALL SYMAMU (N,E,G,D)
      CALL SEARCH (N,X,Y,G,H,D,FO,F,IND,O)
      IF(IND.GT.0) GO TO 240
      FO=F
      DO 210 I=1,N
      X(I)=Y(I)
      C2=H(I)
      Y(I)=TRIAL*D(I)
      H(I)=H(I)-G(I)
  210 G(I)=C2
      CALL SYMAMU(N,E,H,D)
      C1=0.DO
      C2=0.DO
      DO 220 I=1,N
      C1=C1+H(I)*Y(I)
  220 C2=C2-H(I)*D(I)
      DIFF=C1-C2
      SUM=(C1+C2)/(C1*C1)
      C1=1.DO/C1
      C2=1.DO/C2
      L=0
      IF(DIFF)228,222,222
  222 DO 223 I=1,N
  223 H(I)=-C1*D(I)
      DO 225 I=1,N
      V1=SUM*Y(I)-H(I)
      DO 225 J=1,I
      L=L+1
  225 E(L)=E(L)+V1*Y(J)-H(J)*Y(I)
      GO TO 300
  228 DO 230 I=1,N
      V1=C1*Y(I)
      V2=C2*D(I)
      DO 230 J=1,I
      L=L+1
  230 E(L)=E(L)+V1*Y(J)-V2*D(J)
      GO TO 300
  240 IF(IND.GT.2) GO TO 330
      IF(F.GT.FO) GO TO 260
      FO=F
      DO 250 I=1,N
      X(I)=Y(I)
  250 G(I)=H(I)
      GO TO 300
  260 DO 270 I=1,N
  270 D(I)=-G(I)
      CALL SEARCH(N,X,Y,G,H,D,FO,F,IND,O)
      IF(IND.GT.0) GO TO 330
      FO=F
```

```
      DO 280 I=1,N
      X(I)=Y(I)
  280 G(I)=H(I)
  300 CONTINUE
      IND=4
      GO TO 330
  320 ITER=ITER-1
  330 IF(D)RETURN
      WRITE(6,15)ITER
      RETURN
   11 FORMAT (1H1,10X,  'BEHAVIOR UNDER FLETCHER-POWELL ITERATIONS (WITH
     1 FLETCHER UPDATE MODIFICATION)'////
     2             11X,4HITER,2X,3HTRY,4X,8HABSCISSA,10X,5HSLOPE,13X,
     38HFUNCTION ,12X,6HUPDATE)
   15 FORMAT(1HO,10X,'NUMBER OF FLETCHER-POWELL (WITH FLETCHER UPDATA MO
     1DIFICATION) ITERATIONS = ',I3)
      END
      SUBROUTINE FCTGR(X,F,G)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(C)
      INTEGER P
      COMMON/MN/Q1,Q2,A(400),T(100),AT(400),PP(100),Z(100),P,M,N
      COMMON/FUNC/CONST,MM,MM1,ITS,ORT,CCNT(400)
      DIMENSION X(1),G(1),DH(50),DL(50)
      EQUIVALENCE(DH,Z),(DL,Z(51))
      IF(.NOT.ORT)GO TO 50
C ******* ORTHOGONAL ROTATION
C *** COMPUTE T AS A FUNCTION OF X
    1 L=0
      DO 5 I=1,M
      IM=(I-1)*M
      DO 5 J=1,I
      IJ=IM+J
      JI=(J-1)*M+I
      IF(I.EQ.J)GO TO 2
      L=L+1
      T(IJ)=X(L)
      T(JI)=-X(L)
      GO TO 5
    2 T(IJ)=1.DO
    5 CONTINUE
      CALL IFMSL(M,T,PP,G,DET)
      IF(DET.NE.0.DO)GO TO 7
      CALL REORDX(X,ORT)
      GO TO 1
    7 L=0
      DO 10 I=1,M
      DO 10 J=1,M
      L=L+1
      T(L)=2.DO*PP(L)
      IF(I.EQ.J)T(L)=T(L)-1.DO
   10 CONTINUE
C *** COMPUTE F(X)
      CALL MMSL(P,M,M,A,T,AT,0,0,1)
      IF(ITS.EQ.0)GO TO 106
      SUM=0.DO
      SUM1=0.DO
      DO 25 J=1,M
      L=J
      IJ=J
      DQ=0.DO
      DO 20 I=1,P
      DQ=DQ+AT(L)**2
      SUM=SUM+AT(L)**4
   20 L=L+M
      SUM1=SUM1+DQ**2
      DQ=Q2*DQ
```

```
      DO 22 I=1,P
      AT(IJ)=(DO-AT(IJ)**2)*AT(IJ)
   22 IJ=IJ+M
   25 CONTINUE
      F=CONST+Q2*SUM1-SUM
C *** COMPUTE DF/DX=TR(DF/DT'*DT/DX)
      CALL MMSL(M,P,M,AT,A,Z ,1,0,1)
      DO 30 I=1,MM
      Z (I)=4.DO* Z(I)
   30 CONTINUE
      CALL MMSL(M,M,M,PP,Z,AT,0,0,1)
   35 CALL MMSL(M,M,M,AT,PP,Z,0,0,1)
      L=0
      DO 40 I=2,M
      IM=(I-1)*M
      I1=I-1
      DO 40 J=1,I1
      L=L+1
      IJ=IM+J
      JI=(J-1)*M+I
   40 G(L)=2.DO*(Z(IJ)-Z(JI))
      RETURN
C ******* OBLIQUE ROTATION
C *** COMPUTE T AS A FUNCTION OF X
   50 L=0
      K=0
      DO 55 I=1,M
      DO 55 J=1,M
      L=L+1
      IF(I.EQ.J)GO TO 52
      K=K+1
      AT(L)=X(K)
      GO TO 55
   52 AT(L)=1.DO
   55 CONTINUE
      CALL IFMSL(M,AT,T,G,DET)
      IF(DET.NE.0.DO) GO TO 56
      WRITE(6,1000)
      CALL EXIT
   56 L=0
      DO 60 I=1,M
      G(I)=0.DO
      DO 58 K=1,M
      L=L+1
      G(I)=G(I)+T(L)**2
   58 CONTINUE
      G(I)=DSQRT(G(I))
   60 CONTINUE
      L=0
      DO 65 I=1,M
      DO 65 J=1,M
      L=L+1
      PP(L)=T(L)/G(I)
      T(L)=AT(L)*G(J)
   65 CONTINUE
      CALL MMSL(P,M,M,A,T,AT,0,0,1)
      IF(ITS.EQ.0)GO TO 106
      DO 68 J=1,M
   63 DL(J)=0.DO
      SUM1=0.DO
      SUM2=0.DO
      L=0
      DO 75 I=1,P
      SUM=0.DO
      DO 70 J=1,M
      L=L+1
```

```
      TMP=AT(L)**2
      SUM=SUM+TMP
      DL(J)=DL(J)+TMP
      SUM1=SUM1+TMP**2
   70 CONTINUE
      DH(I)=SUM*Q1
   75 SUM2=SUM2+SUM**2
      SUM=0.D0
      DO 80 J=1,M
      L=J
      SUM=SUM+DL(J)**2
      DL(J)=Q2*DL(J)
      DO 80 I=1,P
      AT(L)=AT(L)*(DH(I)+DL(J)-AT(L)**2)
      L=L+M
   80 CONTINUE
      F=Q1*SUM2+Q2*SUM-SUM1
      CALL MMSL(M,P,M,A,AT,Z,1,0,1)
      DO 85 I=1,MM
   85 Z(I)=Z(I)*4.D0
C *** COMPUTE T**-1' DIAG(Z'T)
   86 L=0
      DO 90 I=1,M
      SUM=0.D0
      K=I
      DO 87 J=1,M
      SUM=SUM+Z(K)*T(K)
      K=K+M
   87 CONTINUE
      DO 90 J=1,M
      L=L+1
      AT(L)=PP(L)*SUM
   90 CONTINUE
      CALL MMSL(M,M,M,PP,PP,AT(MM1),0,1,0)
      CALL MMSL(M,M,M,AT,AT(MM1),PP,1,-1,1)
      L=0
      DO 95 I=1,M
      DO 95 J=1,M
      L=L+1
      Z(L)=(Z(L)-PP(L))*G(J)
   95 CONTINUE
      L=0
      K=0
      DO 97 I=1,M
      DO 97 J=1,M
      K=K+1
      IF(I.EQ.J)GO TO 97
      L=L+1
      G(L)=Z(K)
   97 CONTINUE
      RETURN
  106 F=0.D0
      L=0
      DO 110 J=1,P
      DO 110 I=1,M
      L=L+1
      IF(.NOT.CONT(L))GO TO 110
      F=F+AT(L)**2
  110 CONTINUE
      L=0
      DO 120 I=1,M
      DO 120 J=1,M
      L=L+1
      Z(L)=0.D0
      DO 115 K=1,P
      IJM=(K-1)*M
```

```
      IJ=IJM+J
      IF(.NOT.CONT(IJ))GO TO 115
      KI=IJM+I
      KJ=IJM+J
      Z(L)=Z(L)+A(KI)*AT(KJ)
  115 CONTINUE
      Z(L)=2.D0*Z(L)
  120 CONTINUE
      IF(.NOT.ORT)GO TO 86
      CALL MMSL(M,M,M,PP,Z,AT,0,1,1)
      GO TO 35
 1000 FORMAT('OT IS SINGULAR')
      END
      SUBROUTINE SEARCH(N,X,Y,G,H,D,FO,F,IND,O)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      COMMON /FSP/ TRIAL,FACT,AA,BB,EPS,DIFF,MAXTRY,ITER,IP
      DIMENSION X(1),Y(1),G(1),H(1),D(1),P0(3),P1(3),P2(3),P3(3),P4(3)
      IDF=0
      IND=0
      SUM=0.D0
      DO 210 I=1,N
  210 SUM=SUM+D(I)**2
      SUM=DSQRT(SUM)
      IF(N.EQ.1)GO TO 212
      IF(ITER.LE.N)GO TO 212
      FACT=.5D0
      TRIAL=SUM
  212 SUM=1.D0/SUM
      DO 220 I=1,N
  220 D(I)=SUM*D(I)
      SUM=0.D0
      DO 230 I=1,N
  230 SUM=SUM+G(I)*D(I)
      IF(SUM.LT.0.D0)GO TO 240
      IND=3
      RETURN
  240 P0(1)=0.
      P0(2)=SUM
      P0(3)=FO
      DO 250 I=1,3
      P2(I)=P0(I)
  250 P1(I)=P0(I)
      IPS=0
      P4(1)=TRIAL
      ICOUNT=0
      IF(IP.EQ.1)GO TO 255
      IF(ITER.EQ.1)GO TO 255
      IF( O ) WRITE (6,11) ITER,ICOUNT,(P0(I),I=1,3),DIFF
      GO TO 258
  255 IF(O)WRITE(6,11)ITER,ICOUNT,(P0(I),I=1,3)
  258 DO 310 ICOUNT=1,MAXTRY
      DO 260 I=1,N
  260 Y(I)=X(I)+P4(1)*D(I)
      CALL FCTGR  (Y,F,H)
      CALL TIRP(IND)
      IF(IND-1) 265,280,320
  265 SX=P4(1)
      IDF=1
      P4(3)=F
      SUM=0.D0
      DO 270 I=1,N
  270 SUM=SUM+H(I)*D(I)
      P4(2)=SUM
      IF( O ) WRITE (6,12) ICOUNT,(P4(I),I=1,3)
      CALL POINT (P0,P1,P2,P3,P4,IPS)
      IF(IPS.EQ.2) RETURN
```

```
      GO TO 310
  280 IF( J ) WRITE (6,12) ICOUNT,P4(1)
      IF(IDF.GT.0) GO TO 290
      P4(1)=0.500*P4(1)
      GO TO 310
  290 DO 300 I=1,N
  300 Y(I)=X(I)+SX*D(I)
      CALL FCTGR  (Y,F,H)
      RETURN
  310 CONTINUE
      IND=2
  320 RETURN
   11 FORMAT (1H0,9X,2I5,4(3X,D15.8))
   12 FORMAT (15X,I5,3(3X,D15.8))
      END
      SUBROUTINE RECRDX(X,ORT)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      INTEGER P
      COMMON/MN/Q1,Q2,A(400),T(100),AT(400),PP(100),Z(100),P,M,N
      DIMENSION X(1)
      IF (ORT)GO TO 12
      L=0
      K=0
      DO 5 I=1,M
      DO 5 J=1,M
      L=L+1
      IF(I.EQ.J)GO TO 2
      K=K+1
      T(L)=X(K)
      GO TO 5
    2 T(L)=1.D0
    5 CONTINUE
   12 DO 20 I=1,M
      IM=(I-1)*M
      II=IM+I
      DO 20 J=1,M
      IF(I.EQ.J)GO TO 20
      JM=(J-1)*M
      IJ=IM+J
      JI=JM+I
      JJ=JM+J
      IF((DABS(T(IJ))*DABS(T(JI))).LE.(DABS(T(II))*DABS(T(JJ))))GO TO 20
      IK=I
      JK=J
      DO 15 K=1,M
      TMP=T(IK)
      T(IK)=T(JK)
      T(JK)=TMP
      IK=IK+M
      JK=JK+M
   15 CONTINUE
   20 CONTINUE
      L=0
      IF (ORT)GO TO 30
      K=0
      DO 25 I=1,M
      DO 25 J=1,M
      K=K+1
      LL=(J-1)*M+J
      IF(I.EQ.J)GO TO 25
      L=L+1
      X(L)=T(K)/T(LL)
   25 CONTINUE
      RETURN
   30 JJ=1
      DO 40 J=1,M
```

```
      IF(T(JJ).GE.0.D0)GO TO 37
      DO 35 I=1,M
      IJ=(I-1)*M+J
   35 T(IJ)=-T(IJ)
   37 T(JJ)=T(JJ)+1.D0
      JJ=JJ+M+1
   40 CONTINUE
      CALL IFMSL(M,T,T,X,DET)
      IF(DET.NE.0.D0)GO TO 45
      WRITE(6,1000)
      CALL EXIT
   45 L=0
      DO 50 I=2,M
      I1=I-1
      IM=(I-1)*M
      DO 50 J=1,I1
      L=L+1
      IJ=IM+J
   50 X(L)=2.D0*T(IJ)
      RETURN
 1000 FORMAT(1H0,'PROCEDURE BREAKS DOWN - TRY ANOTHER STARTING POINT')
      END
      SUBROUTINE STEDE (N,X,Y,G,H,D,F,EP,C,IND)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      LOGICAL LAV
      COMMON /ESP/ TRIAL,FACT,AA,BB,EPS,DIFF,MAXTRY,ITER,IP
      DIMENSION X(1),Y(1),G(1),H(1),D(1)
      IP=1
      IND=0
      MAXTRY=10
      PR=.5D-1
      TRIAL=.1D0
      FACT=.1D0
      AA=.2D+1
      BB=.75D0
      EPS=EP
      CALL FCTGR  (X,F0,G)
      IF( O ) WRITE (6,11)
      NOCH=0
      DO 200 ITER=1,N
      IF(LAV(N,G,EPS)) GO TO 240
      IF(NOCH.GE.2) GO TO 230
      DO 210 I=1,N
  210 D(I)=-G(I)
      CALL SEARCH (N,X,Y,G,H,D,F0,F,IND,O)
      IF(IND.GT.2.OR.F.GT.F0) GO TO 270
      W=(F0-F)/F0
      F0=F
      DO 220 I=1,N
      X(I)=Y(I)
  220 G(I)=H(I)
      NOCH=NOCH+1
      IF(W.GT.PR) NOCH=0
  200 CONTINUE
  230 IND=4
      GO TO 270
  240 IF(ITER.NE.1)GO TO 260
      WRITE(6,12)
      RETURN
  260 ITER=ITER-1
  270 IF(O)RETURN
      WRITE(6,15)ITER
      RETURN
   12 FORMAT('0AT THE STARTING POINT THE FUNCTION IS AT AN EXACT MINIMUM
     1 OR MAXIMUM'/'0TRY ANOTHER STARTING POINT')
   11 FORMAT (1H1,10X,42HBEHAVIOR UNDER STEEPEST DESCENT ITERATIONS ////
```

```
     1              11X,4HITER,2X,3HTRY,4X,8HABSCISSA,10X,5HSLOPE,13X,
     18HFUNCTION )
  15 FORMAT(1H1,10X,'NUMBER OF STEEPEST-DESCENT ITERATIONS = ',I3)
     END
     SUBROUTINE POINT (P0,P1,P2,P3,P4,IPS)
     IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
     LOGICAL B1,B2,B3,B4,B5
     COMMON /ESP/ TRIAL,FACT,AA,BB,EPS,DIFF,MAXTRY,ITER,IP
     DIMENSION P0(3),P1(3),P2(3),P3(3),P4(3)
     B1=DABS(P4(2)).LT.FACT*DABS(P0(2))
     B2=P4(3).GT.P0(3)
     B3=P4(2).GT.0.D0
     B4=(B1.OR..NOT.B3).AND.B2
     B5=P4(2).GT.P2(2).AND.P2(2).GE.P1(2)
     IF(.NOT.B4) GO TO 220
     IPS=0
     P4(1)=BB*P4(1)
     DO 210 I=1,3
     P1(I)=P0(I)
 210 P2(I)=P0(I)
     RETURN
 220 IF(.NOT.B1) GO TO 230
     IPS=2
     TRIAL=P4(1)
     RETURN
 230 IF(.NOT.B3) GO TO 250
     IPS=1
     DO 240 I=1,3
 240 P3(I)=P4(I)
     CALL INTPOL(P2,P3,P4)
     RETURN
 250 DO 260 I=1,3
     P1(I)=P2(I)
 260 P2(I)=P4(I)
     IF(IPS.NE.1) GO TO 270
     CALL INTPOL(P2,P3,P4)
     RETURN
 270 IF(.NOT.B5) GO TO 280
     CALL INTPOL(P1,P2,P4)
     RETURN
 280 P4(1)=AA*P4(1)
     RETURN
     END
     SUBROUTINE CLEAN(P,M,AT,T,D,PP,E,Y,Z,ORT)
     IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
     INTEGER P
******* E WILL CONTAIN REORDERED AT
******* PP WILL CONTAIN REORDERED T
     DIMENSION AT(1),T(1),D(1),PP(1),E(1),L(10),Y(1),Z(1)
     MP=M*P
     DO 2 I=1,P
   2 Y(I)=0.D0
     DO 20 I=1,M
     D(I)=0.D0
     SUM=0.D0
     JI=I
     L(I)=I
     DO 5 J=1,P
     SUM=SUM+AT(JI)
     D(I)=D(I)+AT(JI)**2
     Y(J)=Y(J)+AT(JI)**2
     JI=JI+M
   5 CONTINUE
     IF(SUM)10,20,20
  10 JI=I
     KK=(I*(I-1))/2+MP
```

```
      DO 14 K=1,M
      IF(K.GT.I)GO TO 12
      KK=KK+1
      GO TO 13
   12 KK=KK+K-1
   13 E(KK)=- E(KK)
      IF(K.EQ.I) E(KK)=-E(KK)
      JI=JI+M
      T(JI)=-T(JI)
   14 CONTINUE
      JI=I
      DO 15 J=1,P
      AT(JI)=-AT(JI)
      JI=JI+M
   15 CONTINUE
   20 CONTINUE
      IF(ORT)GO TO 24
      DO 23 I=1,P
      SUM=0.D0
      KK=(I-1)*M
      DO 22 J=2,M
      JI=J-1
      IJ=KK+J
      M1=JI*J/2+MP
      DO 22 K=1,JI
      M1=M1+1
      IM=KK+K
   22 SUM=SUM+AT(IJ)*AT(IM)*E(M1)
   23 Y(I)=Y(I)+2.D0*SUM
C ******* SORT SUM OF SQUARES OF THE COLUMNS OF AT
   24 N=M
   25 N=N/2
      IF(N.EQ.0)GO TO 55
      KK=M-N
      J=1
   30 I=J
   35 IM=I+N
   40 IF(D(I).GT.D(IM))GO TO 50
   45 J=J+1
      IF(J-KK)30,30,25
   50 TEMP=D(I)
      D(I)=D(IM)
      D(IM)=TEMP
      IJ=L(I)
      L(I)=L(IM)
      L(IM)=IJ
      I=I-N
      IF(I-1)45,35,35
C ******* REORDER AT AND T
   55 M1=M+1
      II=0
      DO 70 I=1,M
      II=M1-I
      N=L(II)
      KK=I
      DO 60 J=1,P
      E(KK)=AT(N)
      KK=KK+M
   60 N=N+M
      N=L(II)
      KK=I
      DO 65 J=1,M
      PP(KK)=T(N)
      N=N+M
      KK=KK+M
   65 CONTINUE
```

```
      IF(ORT)GO TO 70
      N=L(II)
      DO 69 J=1,I
      LI=N
      JJ=M1-J
      II=II+1
      LJ=L(JJ)
      IF(LI.GE.LJ)GO TO 68
      LI=LJ
      LJ=N
   68 KK=((LI-1)*LI)/2+LJ+MP
   69 Z(II)=E(KK)
   70 CONTINUE
      N=M/2
      DO 75 I=1,N
      II=M1-I
      TEMP=D(I)
      D(I)=D(II)
      D(II)=TEMP
   75 CONTINUE
      RETURN
      END
      SUBROUTINE PMSL(N,K,A,FMT,TEXT,LF,LT,IND)
C ***** PRINT MATRIX STORED LINEARLY
C          N,K    ORDER OF MATRIX,I.E. A(NXK)
C          A      MATRIX TO BE PRINTED
C          FMT    VARIABLE FORMAT WITH WHICH A IS PRINTED,SPECIFIED IN THE CALLING
C                 PROGRAM THROUGH DATA CARD OR THE LIKE
C          TEXT   HOLLERITH TITLE OF MATRIX,NUMBER OF CHARACTERS IN THIS TITLE
C                 SHOULD BE A MULTIPLE OF 4
C          LF     CARRIAGE CONTROL DIGIT (I.E. LF=1 IMPLIES NEW PAGE,LF=0 IMPLIES
C                 DOUBLE SPACE  ETC)
C          LT     NUMBER OF WORDS IN TEXT,I.E. NUMBER OF CHARACTERS IN TEXT
C                 DIVIDED BY 4,NOT TO BE EXCEEDED BY 20  (SEE DIMENSION)
C          IND    =0 IMPLIES PRINT FULL MATRIX
C                 OTHERWISE PRINT SYMMETRIC MATRIX,I.E. ONLY LOWER TRIANGULAR PART
C ***********************************
      IMPLICIT REAL*8(A-H,P-Z)
      REAL TEXT,FMT
      DIMENSION A(1),TEXT(20),FMT(1)
      LC=LF
      LO=1
      LL=1
    1 WRITE(6,11)LC,(TEXT(I),I=1,LT)
      L=MINO(LO+9,K)
      WRITE(6,12)(I,I=LO,L)
      IF(IND.EQ.0)GO TO 2
      LL=LO
    2 DO 4 I=LL,N
      IF(IND.EQ.0)GO TO 3
      LCX=(I*(I-1))/2
      LOW=LCX+LO
      LR=LCX+MINO(I,L)
      GO TO 4
    3 LCX=(I-1)*K
      LOW=LCX+LO
      LR=LCX+L
    4 WRITE(6,FMT)I,(A(J),J=LOW,LR)
      IF(L.EQ.K)RETURN
      LO=LO+10
      LC=0
      GO TO 1
   11 FORMAT(I1,10X,20A4)
   12 FORMAT(1HO,10X,10I11)
      END
      LOGICAL FUNCTION LAV(N,G,EPS)
```

```
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      DIMENSION G(1)
      LAV=.FALSE.
      DO 200 I=1,N
      IF(DABS(G(I)).GT.EPS) RETURN
  200 CONTINUE
      LAV=.TRUE.
      RETURN
      END
      SUBROUTINE ROWSCL(P,M,A,SC,IND)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      INTEGER P
C *** PURPOSE : NORMALIZE ROWS OF A GIVEN MATRIX A(PXM) IF IND=0, SCALE BACK
C *** IF IND=1
      DIMENSION A(1),SC(1)
      K=0
      L=0
      DO 25 I=1,P
      IF(IND.EQ.1)GO TO 15
      SC(I)=0.D0
      DO 5 J=1,M
      L=L+1
      SC(I)=SC(I)+A(L)**2
    5 CONTINUE
      SC(I)=DSQRT(SC(I))
      DO 10 J=1,M
      K=K+1
      A(K)=A(K)/SC(I)
   10 CONTINUE
      GO TO 25
   15 DO 20 J=1,M
      K=K+1
      A(K)= A(K)*SC(I)
   20 CONTINUE
   25 CONTINUE
      RETURN
      END
      SUBROUTINE SYMAMU(N,E,G,D)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      DIMENSION E(1),G(1),D(1)
      DO 15 I=1,N
      SUM=0.D0
      IT=(I*(I-1))/2
      DO 10 J=1,N
      L=IT+J
      IF(I.GE.J)GO TO 10
      L=(J*(J-1))/2+I
   10 SUM=SUM-E(L)*G(J)
   15 D(I)=SUM
      RETURN
      END
      SUBROUTINE TIRP (IND)
      COMMON /TID/ SEC
      T=CLOCK(VAD)
      IF(VAD.GT.SEC) IND=5
      RETURN
      END
      SUBROUTINE INTPOL (P1,P2,P3)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      DIMENSION P1(3),P2(3),P3(3)
      Z=(3.D0/(P2(1)-P1(1)))*(P1(3)-P2(3))+P1(2)+P2(2)
      IF(DABS(P1(2)+P2(2)+2.D0*Z).LT.5.D-6)GO TO 200
      W=Z**2-P1(2)*P2(2)
      IF(W.LE.0.D0)GO TO 200
      W=DSQRT(W)
      P3(1)=P1(1)+(1.D0-((P2(2)+W-Z)/(P2(2)-P1(2)+2.D0*W)))*(P2(1)-P1(1)
```

```
     1)
      RETURN
  200 P3(1)=P1(1)-P1(2)*((P2(1)-P1(1))/(P2(2)-P1(2)))
      RETURN
      END
      SUBROUTINE IFMSL(N,B,A,PIVCT,DET)
C ***** INVERT FULL MATRIX STORED LINEARLY
C          N      ORDER OF MATRIX TO BE INVERTED
C          B      MATRIX TO BE INVERTED,STORED LINEARLY
C          A      B INVERSE, STORED LINEARLY
C          PIVOT INTERNAL DUMMY ARRAY,MUST BE DIMENSIONED IN CALLING PROGRAM BY N
C          DET    =0 IF B IS SINGULAR, = DETERMINANT(B) OTHERWISE
C ***************************
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      DIMENSION A(1),B(1),PIVOT(1),IPIV(50),INDEX(50,2)
      EQUIVALENCE (IROW,JROW),(ICOL,JCOL),(AMAX,T,SWAP)
      NN=N*N
      DO 5 J=1,NN
    5 A(J)=B(J)
      IF(N.EQ.1)GO TO 715
   10 DET=1.DO
   15 DO 20 J=1,N
   20 IPIV(J)=0
   30 DO 550 I=1,N
   40 AMAX=0.DO
   45 DO 105 J=1,N
      IRL=(J-1)*N
   50 IF(IPIV(J)-1) 60,105,60
   60 DO 100 K=1,N
      JK=IRL+K
   70 IF(IPIV(K)-1) 80,100,720
   80 IF(DABS(AMAX)-DABS(A(JK)))  85,100,100
   85 IROW=J
   90 ICOL=K
   95 AMAX=A(JK)
  100 CONTINUE
  105 CONTINUE
      IRL=(IROW-1)*N
      ICL=(ICOL-1)*N
  110 IPIV(ICOL)=IPIV(ICOL)+1
  120 IF(IPIV(ICOL)-1) 720,130,720
  130 IF(IROW-ICOL) 140,260,140
  140 DET=-DET
  150 DO 200 L=1,N
      JK=IRL+L
      KJ=ICL+L
  160 SWAP=A(JK)
  170 A(JK)=A(KJ)
  200 A(KJ)=SWAP
  260 INDEX(I,1)=IROW
  270 INDEX(I,2)=ICOL
      JK=ICL+ICOL
  310 PIVOT(I)=A(JK)
  320 DET=DET*PIVOT(I)
  330 A(JK)=1.DO
  340 DO 350 L=1,N
      JK=ICL+L
  350 A(JK)=A(JK)/PIVOT(I)
  380 DO 550 L1=1,N
  390 IF(L1-ICOL) 400,550,400
  400 IRL=(L1-1)*N
      JK=IRL+ICOL
      T=A(JK)
  420 A(JK)=0.
  430 DO 450 L=1,N
      JK=IRL+L
```

```
      KJ=ICL+L
  450 A(JK)=A(JK)-A(KJ)*T
  550 CONTINUE
  600 DO 710 I=1,N
  610 L=N+1-I
  620 IF(INDEX(L,1)-INDEX(L,2)) 630,710,630
  630 JROW= INDEX(L,1)
  640 JCOL= INDEX(L,2)
  650 DO 705 K=1,N
      KJ=(K-1)*N
      JK=KJ+JROW
      ICL=KJ+JCOL
  660 SWAP=A(JK)
  670 A(JK)=A(ICL)
  700 A(ICL)=SWAP
  705 CONTINUE
  710 CONTINUE
      RETURN
  715 IF(B(1).EQ.0.)GO TO 720
      A(1)=1.D0/B(1)
      DET=A(1)
      RETURN
  720 WRITE (6,1)
      DET=0.
      RETURN
    1 FORMAT (23HOTHE MATRIX IS SINGULAR)
      END
      SUBROUTINE MMSL(M,K,N,A,B,C,IA,IB,IC)
C ***** MULTIPLY MATRICES STORED LINEARLY
C          M,K,N ORDER OF MATRICES,I.E. A(MXK),B(KXN),C(MXN)
C          A,B    MATRICES TO BE MULTIPLIED,STORED AS VECTORS
C          C      AB  (A,B,C MUST BE DIMENSIONED IN CALLING PROGRAM)
C          IA,IB IA=0 , IB=0  IMPLIES  C=AB
C                IA=1 , IB=0  IMPLIES  C=A'B
C                IA=0 , IB=1  IMPLIES  C=AB'
C                IA=1 , IB=1  IMPLIES  C=A'B'
C                IA=-1  IMPLIES A IS SYMMETRIC,ONLY LOWER HALF IS STORED
C                IB=-1  IMPLIES B IS SYMMETRIC,ONLY LOWER HALF IS STORED
C          IC     IC=0   IMPLIES C IS SYMMETRIC,ONLY LOWER HALF IS STORED
C                 OTHERWISE C IS A FULL MATRIX
C ***************************
      IMPLICIT REAL*8(A-H,P-Z)
      DIMENSION A(1),B(1),C(1),LL(100)
      NN=N
      DO 333 L=1,K
  333 LL(L)=(L*(L-1))/2
      DO 17 I=1,M
      IP=I+1
      II=(I-1)*N
      IK=(I-1)*K
      I1=(I*(I-1))/2
      IF(IC.EQ.0)NN=I
      DO 17 J=1,NN
      JP=J+1
      JJ=(J-1)*K
      J1=(J*(J-1))/2
      VW=0.D0
      IF(IA)300,200,100
  300 IF(IB)400,500,600
  400 DO 12 L=1,I
      LI=I1+L
      IF(L-J)11,10,10
   10 IL=LL(L)+J
      GO TO 12
   11 IL=J1+L
   12 VW=VW+A(LI)*B(IL)
```

```
      IF(I .EQ.M)GO TO 700
      DO 410 L=IP,K
      LI=LL(L)+I
      IF(L-J)420,430,430
  430 IL=LL(L)+J
      GO TO 410
  420 IL=J1+L
  410 VW=VW+A(LI)*B(IL)
      GO TO 700
  500 DO 510 L=1,I
      LI=I1+L
      IL=(L-1)*N+J
  510 VW=VW+A(LI)*B(IL)
      IF(I .EQ.M)GO TO 700
      DO 520 L=IP,K
      LI=LL(L)+I
      IL=(L-1)*N+J
  520 VW=VW+A(LI)*B(IL)
      GO TO 700
  600 DO 610 L=1,I
      LI=I1+L
      IL=JJ+L
  610 VW=VW+A(LI)*B(IL)
      IF(I .EQ.M)GO TO 700
      DO 620 L=IP,K
      LI=LL(L)+I
      IL=JJ+L
  620 VW=VW+A(LI)*B(IL)
      GO TO 700
  200 IF(IB)240,250,260
  240 DO 210 L=1,J
      LI=IK+L
      IL=J1+L
  210 VW=VW+A(LI)*B(IL)
      IF(J.EQ.NN)GO TO 700
      DO 211 L=JP,K
      LI=IK+L
      IL=LL(L)+J
  211 VW=VW+A(LI)*B(IL)
      GO TO 700
  250 DO 251 L=1,K
      LI=IK+L
      IL=(L-1)*N+J
  251 VW=VW+A(LI)*B(IL)
      GO TO 700
  260 DO 261 L=1,K
      LI=IK+L
      IL=JJ+L
  261 VW=VW+A(LI)*B(IL)
      GO TO 700
  100 IF(IB)60,70,80
   60 DO 61 L=1,J
      LI=(L-1)*M+I
      IL=J1+L
   61 VW=VW+A(LI)*B(IL)
      IF(J.EQ.NN)GO TO 700
      DO 62 L=JP,K
      LI=(L-1)*M+I
      IL=LL(L)+J
   62 VW=VW+A(LI)*B(IL)
      GO TO 700
   70 DO 71 L=1,K
      LI=(L-1)*M+I
      IL=(L-1)*N+J
   71 VW=VW+A(LI)*B(IL)
      GO TO 700
```

```
80 DO 81 L=1,K
   LI=(L-1)*M+I
   IL=JJ+L
81 VW=VW+A(LI)*B(IL)
700 IF(IC)13,14,13
13 IJ=II+J
   GO TO 17
14 IF(I-J)15,16,16
15 IJ=J1+I
   GO TO 17
16 IJ=I1+J
17 C(IJ)=VW
   RETURN
   END
```

II. Listing of Subroutine ROTATE with Subprograms Using the Polak-Ribiere Gradient Method

```
      SUBROUTINE ROTATE(ITP,MAXITE,ORD,CPR,CRT,OSC,ORS,F,X,D,Y,IND,G,H,
     1SC,FMT)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(C)
      INTEGER P
      REAL CLOCK,T1,T2,VAC
      COMMON/FUNC/CONST,MM,MM1,ICF,ITS,OIN,OCNT(400)
      COMMON/MN/Q1,Q2,A(400),T(100),AT(400),PP(100),Z(100),P,M,N
      DIMENSION       X(1),G(1),Y(1),H(1),D(1),      FMT(1),SC(1)
      CALL SETCLK
      OIN=ORT
      EPS=.1D-4*P
      ICF=1
      ITS=ITP
      MM=M*M
      MM1=MM+1
      IF(ITP.EQ.0)GO TO 45
      GO TO(5,10,15,20,25,1),ITP
    1 TMP=Q1+Q2
      Q1=Q1/TMP
      Q2=Q2/TMP
      GO TO 30
    5 Q1=P
      Q2=0.D0
      GO TO 1
   10 Q1=P-1
      Q2=1.D0
      GO TO 1
   15 Q1=2*P-M
      Q2=M
      GO TO 1
   20 Q1=P-1
      Q2=M-1
      GO TO 1
   25 Q1=0.D0
      Q2=P
      GO TO 1
   30 N=M*(M-1)
      IF(CRT)N=N/2
      IF(ORS)GO TO 41
      IF(OSC)CALL ROWSCL(P,M,A,SC,D)
      IF(ORD)GO TO 40
      DO 35 I=1,N
   35 X(I)=0.D0
      GO TO 45
   40 READ(5,100)(FMT(I),I=1,10)
      READ(5,FMT)(X(I),I=1,N)
      GO TO 45
   41 L=0
      K=0
      DO 43 I=1,M
      DO 43 J=1,M
      K=K+1
      LL=(J-1)*M+J
      IF(I.EQ.J)GO TO 43
      L=L+1
      X(L)=T(K)/T(LL)
   43 CONTINUE
   45 T1=CLOCK(VAC)
      IF(MAXITE.EQ.0)MAXITE=11*N
      IF(ITP.EQ.0)GO TO 65
      IF(.NOT.ORT) GO TO 60
      L=0
      CONST=0.D0
```

```
      DO 55 I=1,P
      SUM=0.D0
      DO 50 J=1,M
      L=L+1
   50 SUM=SUM+A(L)**2
   55 CONST=CONST+SUM**2
      CONST=CONST*Q1
      GO TO 65
   60 CALL REORDX(X,IR,ORT)
   65 CALL STEPE(N,X,Y,G,H,D,F,EPS,OPR,IND)
      IF(IND.NE.4)GO TO 75
      CALL POLRIB(N,X,Y,G,H,D,  F,EPS,OPR,IND,MAXITE)
   75 T2=CLOCK(VAD)
      T1=(T1-T2)*1.E-6
      WRITE(6,200)T1,IND
      IF(.NOT.OPR)GO TO 77
      IF(ITP.NE.0)GO TO 77
      WRITE(6,400)
      DO 76 I=1,P
      K=(I-1)*M+1
      J=K+M-1
      WRITE(6,500)(OCNT(L),L=K,J)
   76 CONTINUE
   77 IF(IND.GE.4) WRITE(6,300)(X(I),I=1,N)
      IF(ORT)GO TO 82
      L=MM
      K=0
      DO 80 I=1,M
      DO 80 J=1,I
      L=L+1
      K=K+1
   80 PP(K)=AT(L)
   82 CALL MMSL(P,M,M,A,T,AT,0,0,1)
      IF(OSC)CALL ROWSCL(P,M,AT,SC,1)
      CALL CLEAN(P,M,AT,T,D,PP,  Y,Z,ORT)
      RETURN
  100 FORMAT(10A8)
  200 FORMAT(1H0,10X,'TIME=',F8.2//   11X,'IND=',I1)
  300 FORMAT(1H1,10X,'X=',7D15.7/13X,(7D15.7))
  400 FORMAT(1H1,10X,'LOCATION OF LARGE(F) AND SMALL(T) ELEMENTS OF MATR
     1IX AT')
  500 FORMAT(1H0,10X,40L2/(11X,40L2))
      END
      SUBROUTINE POLRIB(N,X,Y,G,H,D,F,EP,C,IND,MAXITE)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      LOGICAL LAV
      COMMON/ESP/TRIAL,FACT,AA,BB,EPS,DN,MAXTRY,ITER
      COMMON/FUNC/CONST,MM,MM1,ICF,ITS,ORT,OCNT(400)
      DIMENSION X(1),Y(1),G(1),H(1),D(1)
      ITC=0
      IND=0
      EPS=EP
      MAXTRY=10
      TRIAL=.1D0
      FACT=.1D0
      AA=2.D0
      BB=.75D0
      N1=N+1
      MAX=MAXITE/N1+1
      CALL FCTGR(X,F0,G)
      SUMB=0.D0
      DO 2 I=1,N
    2 SUMB=SUMB+G(I)**2
      IF(O)WRITE(6,100)
      DO 25 IT=1,MAX
      IF(LAV(N,G,EPS))GO TO 27
```

```
      CALL FCNRDX(X,IR,ORT)
      IF(IR.EQ.0)GO TO 4
      CALL FCTGR(X,FO,G)
      SUMB=0.D0
      DO 3 I=1,N
    3 SUMB=SUMB+G(I)**2
    4 IF(O)WRITE(6,200)IT
      DO 5 I=1,N
    5 D(I)=-G(I)
      DO 20 ITER=1,N1
      IF(LAV(N,G,EPS))GO TO 28
      CALL SEARCH(N,X,Y,G,H,D,FO,F,IND,O)
      FO=F
      SUM=0.D0
      SUB=0.D0
      DO 10 I=1,N
      SUB=SUB+(H(I)-G(I))*H(I)
   10 SUM=SUM+H(I)**2
      C1=SUB/SUMB*DN
      DO 15 I=1,N
      D(I)=-H(I)+C1*D(I)
      X(I)=Y(I)
   15 G(I)=H(I)
      IF(IND.EQ.0)GO TO 20
      GO TO(30,16,18,30,30),IND
   16 IF(ITC.EQ.1)GO TO 30
      ITC=1
      GO TO 25
   18 IF(ITER.EQ.1)GO TO 30
      GO TO 25
   20 SUMB=SUM
   25 CONTINUE
      IND=4
      GO TO 30
   27 ITER=1
   28 ITER=ITER-1
   30 ITER=(IT-1)*N1+ITER
      IF(.NOT.O  )WRITE(6,300)ITER
      RETURN
  100 FORMAT(1H1,4X,'BEHAVIOR UNDER POLAK-RIBIERE ITERATIONS'////)
  200 FORMAT(5X,'STEP',I3//11X,'ITER',2X,'TRY',4X,'ABSCISSA',10X,'SLOPE'
     1,13X,'FUNCTION')
  300 FORMAT(1HO,10X,'NUMBER OF POLAK-RIBIERE ITERATIONS = ',I3)
      END
      SUBROUTINE FCTGR(X,F,G)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      INTEGER P
      COMMON/MN/Q1,Q2,A(400),T(100),AT(400),PP(100),Z(100),P,M,N
      COMMON/FUNC/CONST,MM,MM1,ICF,ITS,ORT,OCNT(400)
      DIMENSION X(1),G(1),DH(50),DL(50)
      EQUIVALENCE(DH,Z),(DL,Z(51))
      IF(.NOT.ORT)GO TO 50
C *******  ORTHOGONAL ROTATION
C *** COMPUTE T AS A FUNCTION OF X
      DO  5 I=1,MM
    5 T(I)=0.D0
      L=1
      DO 10 I=1,M
      T(L)=1.D0
   10 L=L+M+1
      L=0
      DO 15 J=2,M
      J1=J-1
      DO 15 I=1,J1
      IJ=J-I
      L=L+1
```

```
      XX=X(L)**2
      TMP=1.D0/(1.D0+XX)
      X1=(1.D0-XX)*TMP
      X2=2.D0*X(L)*TMP
      DO 15 K=1,J
      KI=(K-1)*M+I
      KJ=KI+IJ
      TMP=T(KI)
      T(KI)=X1*TMP+X2*T(KJ)
      T(KJ)=-X2*TMP+X1*T(KJ)
   15 CONTINUE
C *** COMPUTE F(X)
      CALL MMSL(P,M,M,A,T,AT,0,0,1)
      IF(ITS.EQ.0)GO TO 93
      SUM=0.D0
      SUM1=0.D0
      DO 25 J=1,M
      L=J
      IJ=J
      DQ=0.D0
      DO 20 I=1,P
      DQ=DQ+AT(L)**2
      SUM=SUM+AT(L)**4
   20 L=L+M
      SUM1=SUM1+DQ**2
      DQ=Q2*DQ
      DO 22 I=1,P
      AT(IJ)=(DQ-AT(IJ)**2)*AT(IJ)
   22 IJ=IJ+M
   25 CONTINUE
      F=CONST+Q2*SUM1-SUM
C *** COMPUTE DF/DX=TR(DF/DT'*DT/DX)
C *** COMPUTE P(0)
      CALL MMSL(M,P,M,AT,A,PP,1,0,1)
      CALL MMSL(M,M,M, T,PP,AT,0,0,1)
      DO 30 I=1,MM
      PP(I)=4.D0*AT(I)
   30 CONTINUE
C ***
   32 X1=1.D0
      X2=0.D0
      IS=1
      JS=2
      L=0
      DO 45 J=2,M
      J1=J-1
      JM=(J-1)*M
      JJ=JM+J
      DO 45 I=1,J1
      IJM=JS-IS
      IM=(I-1)*M
      II=IM+I
      IJ=IM+J
      JI=JM+I
      DO 35 K=1,M
      KI=(K-1)*M+IS
      KJ=KI+IJM
      TMP1=PP(KI)
      PP(KI)=X1*TMP1+X2*PP(KJ)
   35 PP(KJ)=-X2*TMP1+X1*PP(KJ)
      L=L+1
      XX=X(L)**2
      TMP=1.D0/(1.D0+XX)
      X1=(1.D0-XX)*TMP
      X2=2.D0*X(L)*TMP
      IS=I
```

```
      JS=J
      DO 40 K=1,M
      IK=IM+K
      JK=JM+K
      TMP1=PP(IK)
      PP(IK)=X1*TMP1+X2*PP(JK)
   40 PP(JK)=-X2*TMP1+X1*PP(JK)
      G(L)=2.D0*TMP*(-X2*(PP(II)+PP(JJ))+X1*(PP(IJ)-PP(JI)))
   45 CONTINUE
      RETURN
C ******* OBLIQUE ROTATION
C *** COMPUTE T AS A FUNCTION OF X
   50 L=0
      K=0
      DO 55 I=1,M
      DO 55 J=1,M
      L=L+1
      IF(I.EQ.J)GO TO 52
      K=K+1
      AT(L)=X(K)
      GO TO 55
   52 AT(L)=1.D0
   55 CONTINUE
      CALL IFMSL(M,AT,T,G,DET)
      IF(DET.NE.0.D0) GO TO 56
      WRITE(6,1000)
      CALL EXIT
   56 L=0
      DO 60 I=1,M
      G(I)=0.D0
      DO 58 K=1,M
      L=L+1
      G(I)=G(I)+T(L)**2
   58 CONTINUE
      G(I)=DSQRT(G(I))
   60 CONTINUE
      L=0
      DO 65 I=1,M
      DO 65 J=1,M
      L=L+1
      PP(L)=T(L)/G(I)
      T(L)=AT(L)*G(J)
   65 CONTINUE
      CALL MMSL(P,M,M,A,T,AT,0,0,1)
      IF(ITS.EQ.0)GO TO 93
      DO 68 J=1,M
   68 DL(J)=0.D0
      SUM1=0.D0
      SUM2=0.D0
      L=0
      DO 75 I=1,P
      SUM=0.D0
      DO 70 J=1,M
      L=L+1
      TMP=AT(L)**2
      SUM=SUM+TMP
      DL(J)=DL(J)+TMP
      SUM1=SUM1+TMP**2
   70 CONTINUE
      DH(I)=SUM*Q1
   75 SUM2=SUM2+SUM**2
      SUM=0.D0
      DO 80 J=1,M
      L=J
      SUM=SUM+DL(J)**2
      DL(J)=Q2*DL(J)
```

```
      DO 80 I=1,P
      AT(L)=AT(L)*(DH(I)+DL(J)-AT(L)**2)
      L=L+M
   80 CONTINUE
      F=Q1*SUM2+Q2*SUM-SUM1
      CALL MMSL(M,P,M,A,AT,Z,1,C,1)
      DO 85 I=1,MM
   85 Z(I)=Z(I)*4.DO
C *** COMPUTE T**-1* DIAG(Z*T)
   86 L=0
      DO 90 I=1,M
      SUM=0.DO
      K=I
      DO 87 J=1,M
      SUM=SUM+Z(K)*T(K)
      K=K+M
   87 CONTINUE
      DO 90 J=1,M
      L=L+1
      AT(L)=PP(L)*SUM
   90 CONTINUE
      CALL MMSL(M,M,M,PP,PP,AT(MM1),0,1,0)
      CALL MMSL(M,M,M,AT,AT(MM1),PP,1,-1,1)
      L=0
      DO 91 I=1,M
      DO 91 J=1,M
      L=L+1
      Z(L)=(Z(L)-PP(L))*G(J)
   91 CONTINUE
      L=0
      K=0
      DO 92 I=1,M
      DO 92 J=1,M
      K=K+1
      IF(I.EQ.J)GO TO 92
      L=L+1
      G(L)=Z(K)
   92 CONTINUE
      RETURN
   93 IF(ICF.EQ.0)GO TO 106
      CONST=.2500
      L=0
      LL=0
      DO 97 J=1,P
      L=L+1
      TMP=AT(L)**2
      DO 95 I=2,M
      L=L+1
      TMP1=AT(L)**2
      IF(TMP1.GT.TMP)TMP=TMP1
   95 CONTINUE
      )   =TMP*CONST
      DO 96 I=1,M
      LL=LL+1
      DCNT(LL)=.FALSE.
      IF(AT(LL)**2.GT.0    )GO TO 96
      DCNT(LL)=.TRUE.
   96 CONTINUE
   97 CONTINUE
      ICF=0
  106 F=0.DO
      L=0
      DO 110 J=1,P
      DO 110 I=1,M
      L=L+1
      IF(.NOT.DCNT(L ))GO TO 110
```

```
      F=F+AT(L)**2
  110 CONTINUE
      L=0
      DO 120 I=1,M
      DO 120 J=1,M
      L=L+1
      Z(L)=0.D0
      DO 115 K=1,P
      IJM=(K-1)*M
      IJ=IJM+J
      IF(.NOT.DCNT(IJ))GO TO 115
      KI=IJM+I
      KJ=IJM+J
      Z(L)=Z(L)+A(KI)*AT(KJ)
  115 CONTINUE
      Z(L)=2.D0*Z(L)
  120 CONTINUE
      IF(.NOT.ORT)GO TO 86
  125 CALL MMSL(M,M,M,T,Z,PP, 0,1,1)
      GO TO 32
 1000 FORMAT('OT IS SINGULAR')
      END
      SUBROUTINE SEARCH(N,X,Y,G,H,D,FO,F,IND,O)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      COMMON/ESP/TRIAL,FACT,AA,BB,EPS,DN,MAXTRY,ITER
      DIMENSION X(1),Y(1),G(1),H(1),D(1),PO(3),P1(3),P2(3),P3(3),P4(3)
      IDF=0
      IND=0
      SUM=0.D0
      DO 210 I=1,N
  210 SUM=SUM+D(I)**2
      DN=DSQRT(SUM)
      SUM=1.D0/DN
      DO 220 I=1,N
  220 D(I)=SUM*D(I)
      SUM=0.D0
      DO 230 I=1,N
  230 SUM=SUM+G(I)*D(I)
      IF(SUM.LT.0.D0)GO TO 240
      IND=3
      RETURN
  240 PO(1)=0.
      PO(2)=SUM
      PO(3)=FO
      DO 250 I=1,3
      P2(I)=PO(I)
  250 P1(I)=PO(I)
      IPS=0
      P4(1)=TRIAL
      ICOUNT=0
      IF(O)WRITE(6,11)ITER,ICOUNT,(PO(I),I=1,3)
      DO 310 ICOUNT=1,MAXTRY
      DO 260 I=1,N
  260 Y(I)=X(I)+P4(1)*D(I)
      CALL FCTGR  (Y,F,H)
      CALL TIRP(IND)
      IF(IND-1) 265,230,320
  265 SX=P4(1)
      IDF=1
      P4(3)=F
      SUM=0.D0
      DO 270 I=1,N
  270 SUM=SUM+H(I)*D(I)
      P4(2)=SUM
      IF( O ) WRITE (6,12) ICOUNT,(P4(I),I=1,3)
      CALL POINT (PO,P1,P2,P3,P4,IPS)
```

```
      IF(IPS.EQ.2) RETURN
      GO TO 310
  280 IF( O ) WRITE (6,12) ICOUNT,P4(1)
      IF(IDF.GT.0) GO TO 290
      P4(1)=0.500*P4(1)
      GO TO 310
  290 DO 300 I=1,N
  300 Y(I)=X(I)+SX*D(I)
      CALL FCTGR  (Y,F,H)
      RETURN
  310 CONTINUE
      IND=2
  320 RETURN
   11 FORMAT (1HO,9X,2I5,3(3X,D15.8))
   12 FORMAT (15X,I5,3(3X,D15.8))
      END
      SUBROUTINE REORDX(X,IR,ORT)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      INTEGER P
      COMMON/MN/O1,O2,A(400),T(100),AT(400),PP(100),Z(100),P,M,N
      DIMENSION X(1)
      IF (ORT)GO TO 12
      L=0
      K=0
      DO 5 I=1,M
      DO 5 J=1,M
      L=L+1
      IF(I.EQ.J)GO TO 2
      K=K+1
      T(L)=X(K)
      GO TO 5
    2 T(L)=1.DO
    5 CONTINUE
   12 IR=0
      DO 20 I=1,M
      IM=(I-1)*M
      II=IM+I
      DO 20 J=1,M
      IF(I.EQ.J)GO TO 20
      JM=(J-1)*M
      IJ=IM+J
      JI=JM+I
      JJ=JM+J
      IF((DABS(T(IJ))*DABS(T(JI))).LE.(DABS(T(II))*DABS(T(JJ))))GO TO 20
      IK=I
      JK=J
      DO 15 K=1,M
      TMP=T(IK)
      T(IK)=T(JK)
      T(JK)=TMP
      IK=IK+M
      JK=JK+M
   15 CONTINUE
      IR=1
   20 CONTINUE
      L=0
      IF (ORT)GO TO 30
      K=0
      DO 25 I=1,M
      DO 25 J=1,M
      K=K+1
      LL=(J-1)*M+J
      IF(I.EQ.J)GO TO 25
      L=L+1
      X(L)=T(K)/T(LL)
   25 CONTINUE
```

```
      RETURN
   30 JJ=1
      DO 40 J=1,M
      IF(T(JJ).GE.0.D0)GO TO 37
      IR=1
      DO 35 I=1,M
      IJ=(I-1)*M+J
   35 T(IJ)=-T(IJ)
   37 T(JJ)=T(JJ)+1.D0
      JJ=JJ+M+1
   40 CONTINUE
      CALL IEMSL(M,T,T,X,DET)
      IF(DET.NE.0.D0)GO TO 45
      WRITE(6,1000)
      CALL EXIT
   45 L=0
      DO 50 I=2,M
      I1=I-1
      IM=(I-1)*M
      DO 50 J=1,I1
      L=L+1
      IJ=IM+J
   50 X(L)=2.D0*T(IJ)
      RETURN
 1000 FORMAT(1H0,'PROCEDURE BREAKS DOWN - TRY ANOTHER STARTING POINT')
      END
      SUBROUTINE STEDE (N,X,Y,G,H,D,F,EP,C,IND)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      LOGICAL LAV
      COMMON/ESP/TRIAL,FACT,AA,BB,EPS,DN,MAXTRY,ITER
      DIMENSION X(1),Y(1),G(1),H(1),D(1)
      IND=0
      MAXTRY=10
      PR=.5D-1
      TRIAL=.1D0
      FACT=.1D0
      AA=.2D+1
      BB=.75D0
      EPS=EP
      CALL FCTGR  (X,F0,G)
      IF( O ) WRITE (6,11)
      NOCH=0
      DO 200 ITER=1,N
      IF(LAV(N,G,EPS)) GO TO 240
      IF(NOCH.GE.2) GO TO 230
      DO 210 I=1,N
  210 D(I)=-G(I)
      CALL SEARCH (N,X,Y,G,H,D,F0,F,IND,O)
      IF(IND.GT.2.OR.F.GT.F0) GO TO 270
      W=(F0-F)/F0
      F0=F
      DO 220 I=1,N
      X(I)=Y(I)
  220 G(I)=H(I)
      NOCH=NOCH+1
      IF(W.GT.PR) NOCH=0
  200 CONTINUE
  230 IND=4
      GO TO 270
  240 IF(ITER.NE.1)GO TO 260
      WRITE(6,12)
      RETURN
  260 ITER=ITER-1
  270 IF(O)RETURN
      WRITE(6,15)ITER
      RETURN
```

```
   12 FORMAT('0AT THE STARTING POINT THE FUNCTION IS AT AN EXACT MINIMUM
     1 OR MAXIMUM'/'0TRY ANOTHER STARTING POINT')
   11 FORMAT (1H1,10X,42HBEHAVIOR UNDER STEEPEST DESCENT ITERATIONS ////
     1            11X,4HITER,2X,3HTRY,4X,8HABSCISSA,10X,5HSLOPE,13X,
     18HFUNCTION )
   15 FORMAT(1H1,10X,'NUMBER OF STEEPEST-DESCENT ITERATIONS = ',I3)
      END
      SUBROUTINE POINT (P0,P1,P2,P3,P4,IPS)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      LOGICAL B1,B2,B3,B4,B5
      COMMON/ESP/TRIAL,FACT,AA,BB,EPS,DN,MAXTRY,ITER
      DIMENSION P0(3),P1(3),P2(3),P3(3),P4(3)
      B1=DABS(P4(2)).LT.FACT*DABS(P0(2))
      B2=P4(3).GT.P0(3)
      B3=P4(2).GT.0.D0
      B4=(B1.OR..NOT.B3).AND.B2
      B5=P4(2).GT.P2(2).AND.P2(2).GE.P1(2)
      IF(.NOT.B4) GO TO 220
      IPS=0
      P4(1)=BB*P4(1)
      DO 210 I=1,3
      P1(I)=P0(I)
  210 P2(I)=P0(I)
      RETURN
  220 IF(.NOT.B1) GO TO 230
      IPS=2
      TRIAL=P4(1)
      RETURN
  230 IF(.NOT.B3) GO TO 250
      IPS=1
      DO 240 I=1,3
  240 P3(I)=P4(I)
      CALL INTPOL(P2,P3,P4)
      RETURN
  250 DO 260 I=1,3
      P1(I)=P2(I)
  260 P2(I)=P4(I)
      IF(IPS.NE.1) GO TO 270
      CALL INTPOL(P2,P3,P4)
      RETURN
  270 IF(.NOT.B5) GO TO 280
      CALL INTPOL(P1,P2,P4)
      RETURN
  280 P4(1)=AA*P4(1)
      RETURN
      END
      SUBROUTINE CLEAN(P,M,AT,T,D,PP,  Y,Z,ORT)
      IMPLICIT REAL*8(A-H,P-Z),LOGICAL*1(O)
      INTEGER P
C ******* AT WILL CONTAIN REORDERED AT
C ******* PP WILL CONTAIN REORDERED T
C ******* Z WILL CONTAIN REORDERED FACTOR CORRELATION MATRIX (IN OBLIQUE CASE)
      DIMENSION AT(1),T(1),D(1),PP(1),Z(1),L(10),Y(1)
      DO 2 I=1,P
    2 Y(I)=0.D0
      DO 20 I=1,M
      D(I)=0.D0
      SUM=0.D0
      JI=I
      L(I)=I
      DO 5 J=1,P
      SUM=SUM+AT(JI)
      D(I)=D(I)+AT(JI)**2
      Y(J)=Y(J)+AT(JI)**2
      JI=JI+M
    5 CONTINUE
```

```
      IF(SUM)10,20,20
   10 JI=I
      KK=(I*(I-1))/2
      DO 14 K=1,M
      IF(K.GT.I)GO TO 12
      KK=KK+1
      GO TO 13
   12 KK=KK+K-1
   13 PP(KK)=-PP(KK)
      IF(K.EQ.I)PP(KK)=-PP(KK)
      JI=JI+M
      T(JI)=-T(JI)
   14 CONTINUE
      JI=I
      DO 15 J=1,P
      AT(JI)=-AT(JI)
      JI=JI+M
   15 CONTINUE
   20 CONTINUE
      IF(ORT)GO TO 24
      DO 23 I=1,P
      SUM=0.D0
      KK=(I-1)*M
      DO 22 J=2,M
      JI=J-1
      IJ=KK+J
      M1=JI*J/2
      DO 22 K=1,JI
      M1=M1+1
      IM=KK+K
   22 SUM=SUM+AT(IJ)*AT(IM)*PP(M1)
   23 Y(I)=Y(I)+2.D0*SUM
C ******* SORT SUM OF SQUARES OF THE COLUMNS OF AT
   24 N=M
   25 N=N/2
      IF(N.EQ.0)GO TO 55
      KK=M-N
      J=1
   30 I=J
   35 IM=I+N
   40 IF(D(I).GT.D(IM))GO TO 50
   45 J=J+1
      IF(J-KK)30,30,25
   50 TEMP=D(I)
      D(I)=D(IM)
      D(IM)=TEMP
      IJ=L(I)
      L(I)=L(IM)
      L(IM)=IJ
      I=I-N
      IF(I-1)45,35,35
C ******* REORDER AT AND T
   55 M1=M+1
      IF(ORT)GO TO 59
      I1=0
      DO 58 I=1,M
      II=M1-I
      N=L(II)
      DO 58 J=1,I
      LI=N
      JJ=M1-J
      I1=I1+1
      LJ=L(JJ)
      IF(LI.GE.LJ)GO TO 57
      LI=LJ
      LJ=N
```

```
57 KK=((LI-1)*LI)/2+LJ
   Z(I1)=PP(KK)
58 CONTINUE
59 DO 70 I=1,M
   II=M1-I
   N=L(II)
   KK=I
   DO 70 J=1,M
   PP(KK)=T(N)
   N=N+M
   KK=KK+M
70 CONTINUE
   N=M/2
   DO 75 I=1,N
   II=M1-I
   TEMP=D(I)
   D(I)=D(II)
   D(II)=TEMP
75 CONTINUE
    DO 65 I=1,M
   II=M1-I
   N=L(II)     .
   IF(I.EQ.N)GO TO 65
   KK=I
   DO 60 J=1,P
   TMP=AT(KK)
   AT(KK)=AT(N)
   AT(N)=TMP
   KK=KK+M
60 N=N+M
   DO 62 K=1,M
   N=M1-K
   IF(L(N).EQ.I)L(N)=L(II)
62 CONTINUE
65 CONTINUE
   RETURN
   END
```

Note: Subprograms PMSL, LAV, ROWSCL, SYMAMU, TIRP, INTPOL, IFMSL, and MMSL are not listed here again since they are the same as listed in I above. (See section 4.)